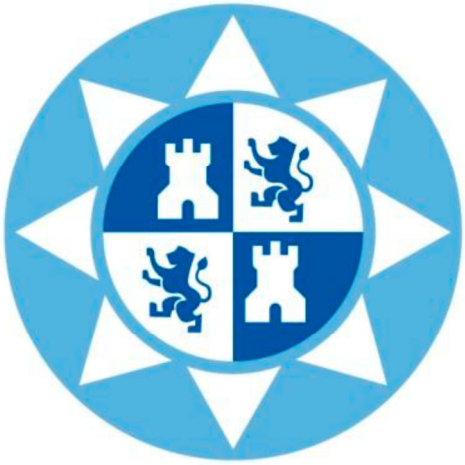
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA



Proyecto Fin de Carrera

¿Dónde como hoy? Una aplicación Android para los miembros de la Upct y Visitantes.

AUTOR: Salvador Bernabé Soto

DIRECTORA: Pilar Manzanares López

Septiembre / 2014

Autor	Salvador Bernabé Soto
E-mail del autor	salva@sbernabe.com , saggiator@gmail.com
Director(es)	Pilar Manzanares López
Título del PFC	¿Dónde como hoy? Una aplicación Android para los miembros de la Upct y Visitantes.

Resumen: El objetivo de este proyecto consiste en desarrollar una aplicación para dispositivos móviles Android (smartphones y tabletas).

La finalidad de esta aplicación es proporcionar a los estudiantes y personal de la UPCT, así como a cualquier persona que pudiera estar interesada, información visual, sencilla y directa sobre los menús diarios ofrecidos en las diferentes cantinas situadas en la UPCT. El usuario podrá decidir entre las diferentes opciones culinarias ofrecidas en los establecimientos sin necesidad de desplazarse para conocer el menú.

Si bien los potenciales comensales son los destinatarios de la aplicación, también serán usuarios de la misma el personal de las cantinas, ya que deberán, a través de la aplicación, aportar la información sobre los menús ofertados.

Titulación	Ingeniero Técnico en Telecomunicación, Esp. Telemática.
Departamento	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación	Septiembre 2014

Índice.

1. Introducción y objetivos.

En las siguientes líneas se hace una breve introducción al presente proyecto, exponiendo cuál es su motivación y cuáles son los objetivos que persiguen.

1.1. Introducción.

El presente documento recoge toda la información del desarrollo de software llevado a cabo para el proyecto final de carrera. Este proyecto surge a partir de la motivación de desarrollar una aplicación para una plataforma móvil que permita al usuario saber los menús ofrecidos por una serie de restaurantes, con la posibilidad de consultar y reservar cada uno de ellos a partir de una fecha y hora determinada. La aplicación desarrollada ha recibido el nombre de “Hoy como en casa”.

Con este objetivo global, se ha utilizado para el desarrollo de la plataforma Android al ser una de las plataformas con mayor ratio de crecimiento del mercado y por permitir mediante interfaces, crear todas las funcionalidades que eran necesarias para el desarrollo de esta aplicación como se podrá ver a lo largo del documento.

1.2. Objetivos.

El objetivo principal de este proyecto es desarrollar una aplicación para dispositivos móviles que facilite al usuario la consulta y reserva de una lista de menús ofrecidos por varios restaurantes, mediante una interfaz simple y sencilla.

Los dispositivos móviles utilizados son smartphones con sistema operativo Android. Este sistema operativo ha sido seleccionado para el presente proyecto por estos motivos:

- Sistema que cuenta con el respaldo de Google, una de las empresas más grandes a nivel mundial.
- Numerosos distribuidores y operadores se han unido a la plataforma de patrocinadores de Android, y fabricantes de gran renombre han anunciado la producción inminente de nuevos modelos con Android como sistema nativo, siendo algunos de ellos: HTC, Samsung, Motorola, etc.
- Crecimiento de ventas de dispositivos con este sistema operativo, llegando incluso a ser líder en algunos países.
- Es posible la realización de aplicaciones en ordenadores con sistema operativo Windows, ampliamente extendido.

- Existencia de un plugins que permite el desarrollo de la aplicación en el entorno de programación Eclipse, con todas las facilidades que ello conlleva.
- El lenguaje de programación utilizado es Java, uno de los lenguajes más conocidos por los programadores.

1.3. Estructura de la Memoria.

La organización de los capítulos de la memoria está compuesta de la siguiente forma:

Capítulo 2: se van a exponer las diferentes tecnologías que se han empleado para el desarrollo de dicha aplicación.

Capítulo 3: se explica cómo se ha preparado el entorno de desarrollo de la aplicación para su implementación, que herramientas han sido necesarias para poder llevarlo a cada y su configuración.

Capítulo 4: se describe el desarrollo de la aplicación y se destacan aspectos de especial interés del desarrollo de la misma

Capítulo 5: en este apartado se explica con detalle el uso de la aplicación para los usuarios, con un manual y una breve descripción de la misma.

Capítulo 6: se ofrecen conclusiones después de la finalización de la aplicación, además de las posibles mejoras para su mejor funcionamiento en un futuro no muy lejano.

Capítulo 7: Bibliografía.

2. Tecnologías Empleadas.

En esta sección vamos a presentar las principales tecnologías que han permitido el desarrollo de la app “Hoy como en casa” basada en el sistema Android.

El lenguaje de programación Java es el lenguaje utilizado para la creación de la aplicación Android. El entorno de programación utilizado para ello es “Eclipse”.

La aplicación requiere del almacenamiento de datos, tanto datos asociados a los menús y restaurantes, como datos asociados a las reservas de los usuarios. Para ello, en este trabajo se ha utilizado el gestor de bases de datos MySQL para almacenar todos los datos de la aplicación.

Otro lenguaje de programación utilizado es PHP, que es el responsable de conectar Android con Mysql. Por otra parte, PHP genera Json, un formato ligero para el intercambio de datos, que permite llevar la información de la BBDD Mysql al terminal Android.

2.1. Java.

El lenguaje de programación Java es el utilizado para la creación de aplicaciones para los dispositivos Android. Es un lenguaje orientado a objetos desarrollado por Sun Microsystems a principios de los años 90 y algunas de sus características principales son:

Sencillez: el lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Robusto: Java fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. La recolección de basura elimina la necesidad de liberación explícita de memoria.

Multihilo: Java soporta sincronización de múltiples hilos de ejecución (multithreading) a nivel de lenguaje, pudiendo así una aplicación ejecutar múltiples acciones a la vez.

Permite realizar aplicaciones distribuidas: Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.

Indiferente a la arquitectura: el problema de los dispositivos móviles, Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos.

2.2. Mysql.

MySQL (cuyas siglas en inglés se corresponden con *My Structured Query Language* o Lenguaje de Consulta Estructurado). En 1980 los programadores de IBM lo desarrollaron para contar con un código de programación que permitiera generar múltiples y extendidas bases de datos para empresas y organizaciones de diferente tipo. Desde esta época numerosas versiones han surgido y muchas de ellas fueron de gran importancia. Hoy en día MySQL es desarrollado por la empresa Sun Mycrosystems.

Una de las características más interesantes de MySQL es que permite recurrir a bases de datos multiusuario a través de la web y en diferentes lenguajes de programación que se adaptan a diferentes necesidades y requerimientos. Por otro lado, MySQL es conocida por desarrollar alta velocidad en la búsqueda de datos e información, a diferencia de sistemas anteriores. Las plataformas que utiliza son de variado tipo y entre ellas podemos mencionar LAMP, MAMP, SAMP, BAMP y WAMP (aplicables a Mac, Windows, Linux, BSD, Open Solaris, Perl y Phyton entre otras).

2.3. PHP.

PHP es un acrónimo recursivo que significa PHP Hypertext Pre-processor. Fue creado originalmente por Rasmus Lerdof en 1994; sin embargo la implementación principal de PHP es producida ahora por “The PHP Group” y sirve cómo el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera está licencia cómo software libre.

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al

igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

Se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy.

2.4. Json.

Json, cuyas siglas son (*JavaScript Object Notation* - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript.

JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En otros lenguajes esto es conocido cómo un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

2.5. Entorno de programación Eclipse.

El entorno de programación Eclipse comenzó como un proyecto de IBM. En la actualidad es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto, es decir, código distribuido y desarrollado libremente.

Consiste en un Entorno de Desarrollo Integrado (IDE, *Integrated Development Environment*). Un IDE es un programa compuesto por un conjunto de herramientas útiles para un desarrollador de software. Como elementos básicos, cuenta con un editor de código, un compilador/intérprete y un depurador.

Eclipse es utilizado mayoritariamente como IDE Java, llamado éste *Java Development Toolkit* (JDT). Aunque también da soporte a otros lenguajes de programación, como son C/C++, Cobol, Fortran, PHP o Python.

A la plataforma base de Eclipse se le pueden añadir complementos (plugins) para extender la funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.

Así pues, para integrar Android en Eclipse, se utiliza el plugin ADT (*Android Development Tools*), que permite el uso de las herramientas y métodos imprescindibles para la realización de aplicaciones. Estos métodos y aplicaciones están recogidas en el SDK (*Software Developer Kit*) de Android, que además disponible una amplia documentación de respaldo.

Para la creación de la app "Hoy como en casa" se han utilizado Eclipse y todos los complementos necesarios para el desarrollo de aplicaciones Android.

2.6. Plataforma Android.

Android es una solución completa de software de código libre para dispositivos móviles, la cual se encuentra instalada en el smartphone donde se han ejecutado las pruebas de este proyecto.

Se distribuye bajo una licencia Apache, versión 2, lo que implica que, al tratarse de software libre, cualquier desarrollador tiene acceso completo al SDK del sistema, incluidas todas sus APIS, documentación y emulador para pruebas, pudiendo distribuirlo y modificarlo. Además, esta licencia permite a los desarrolladores tanto publicar sus creaciones, como distribuirlas ocultando el código fuente.

Android proporciona un paquete completo de software a todos los niveles:

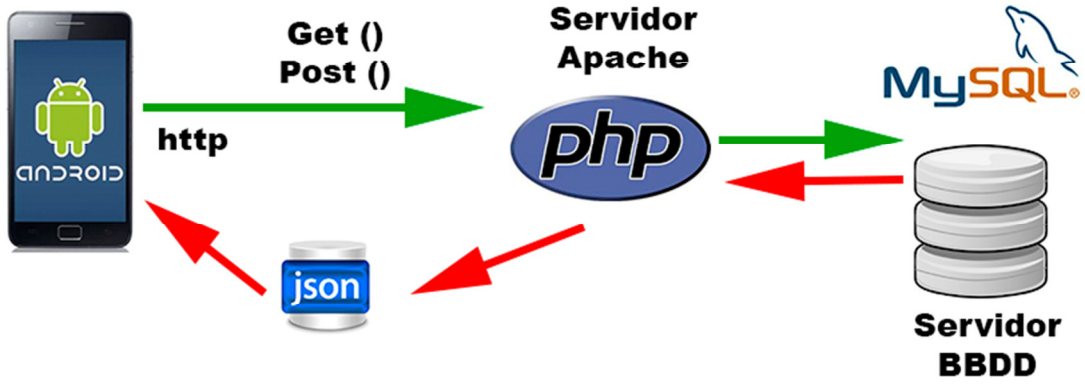
- Un kernel Linux que sirve como base de la pila de software y se encarga de las funciones más básicas del sistema: gestión de drivers, seguridad, comunicaciones, etc.
- Una capa de librerías de bajo nivel en C y C++.
- Un framework para el desarrollo de aplicaciones.
- Una suite de aplicaciones (navegador, agenda, gestión del teléfono).

Las aplicaciones Android están programadas en lenguaje Java, y corren sobre Dalvik, una máquina virtual Java desarrollada por Google.

Con Android se busca reunir en una misma plataforma todos los elementos necesarios que permitan al desarrollador controlar y aprovechar al máximo cualquier funcionalidad ofrecida por un dispositivo móvil (llamadas, mensajes de texto, cámara, agenda de contactos, conexión Wi-Fi, Bluetooth, aplicaciones ofimáticas, videojuegos, etc.), así como poder crear aplicaciones.

2.7. Funcionamiento General de la Aplicación.

A continuación, mostramos un ejemplo del funcionamiento general de la aplicación:



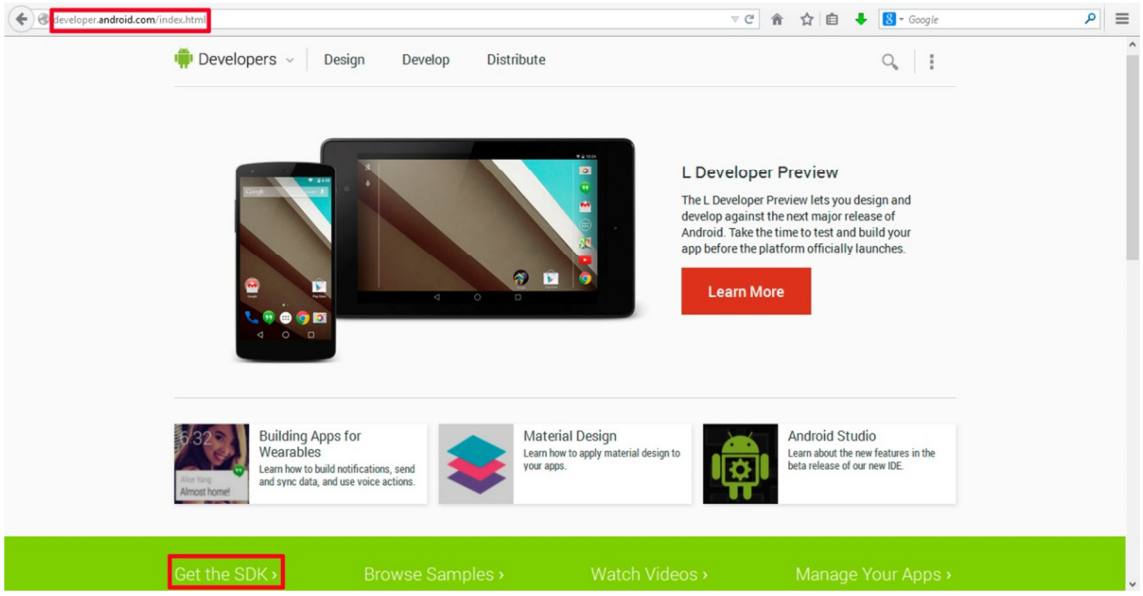
Como podemos observar, la aplicación Android genera una petición http utilizando las funciones “get” o “post” al servidor Apache y este la procesa mediante php. Una vez que ha llegado esta solicitud, php establece conexión a la base de datos con una consulta y se la devuelve al php. Automáticamente php genera un json con la toda la información de la consulta de forma estructurada y la envía al terminal de la aplicación Android.

Se utiliza la función “get” para consultar o extraer y borrar datos de la base de datos. Por el contrario, se utiliza la función “post” para añadir o modificar datos en la base de datos.

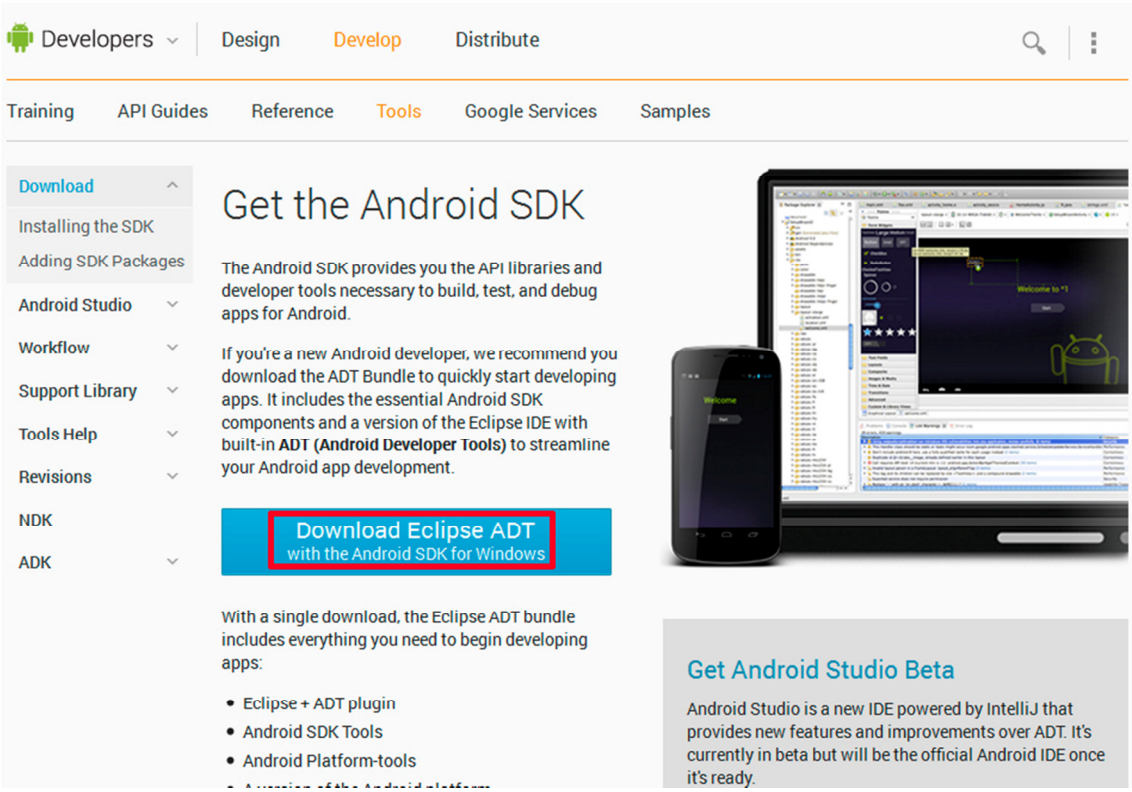
3. Entorno de Desarrollo de la Aplicación.

3.1. Instalación y configuración de SDK para Windows.

En nuestro navegador, escribimos la siguiente dirección: http://developer.android.com/index.html y nos aparecerá la web como se muestra en la siguiente captura, buscamos “Get the SDK” y clicamos sobre él.

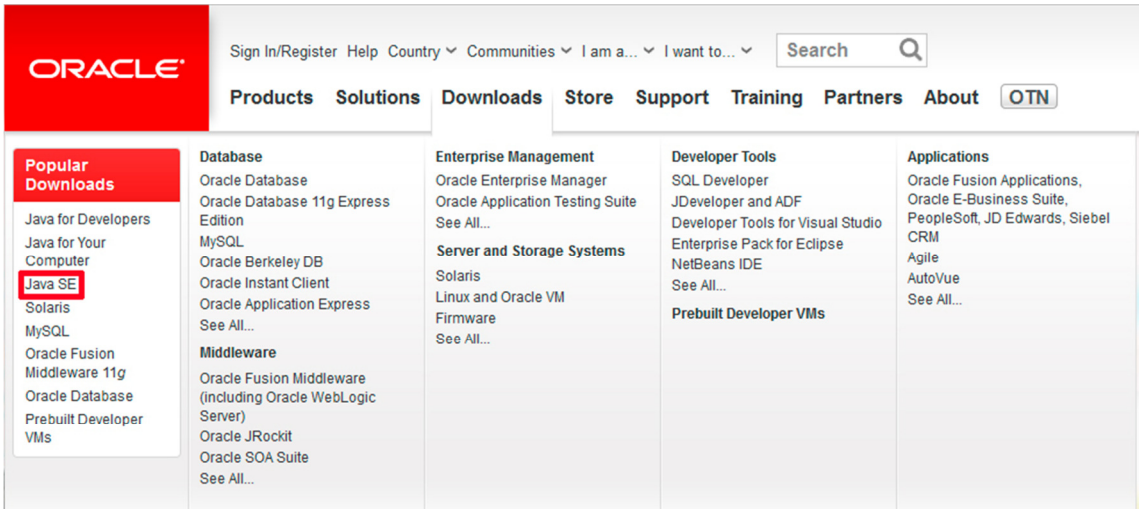


A continuación, debemos descargar “Eclipse ADT”



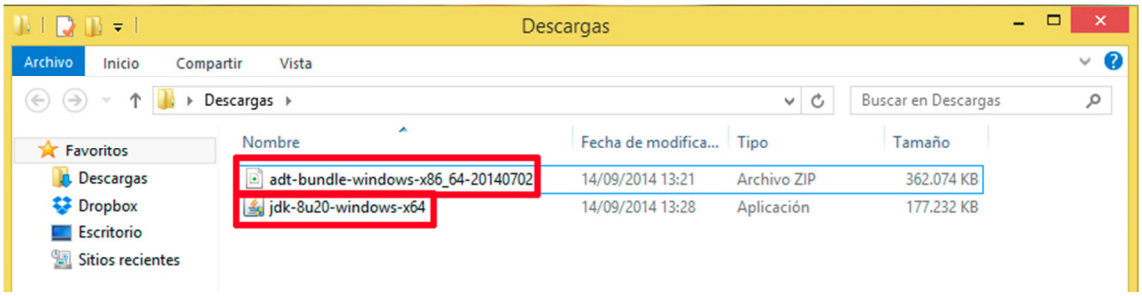
Debemos aceptar las condiciones que nos proporcionan, elegimos la versión de 64 bits para nuestro sistema operativo y procedemos a la descarga del fichero.

Debemos de descargar el Java SDK. Accedemos con nuestro navegador a la siguiente dirección: www.oracle.com/index.php, pulsamos sobre la **"Download"** y en la parte de la izquierda nos aparecerá Java SE y clicamos sobre el cómo se nuestra en la siguiente captura:



Procedemos a descargar la última versión disponible a día de hoy. De igual forma que antes, debemos aceptar los términos para proceder a la descarga del fichero. Seleccionamos la versión de 64 bits para Windows y guardamos el fichero en nuestro equipo para su posterior instalación.

Una vez terminadas las dos descargas, procedemos a la instalación y configuración de las mismas. Para ello debemos tener los siguientes ficheros:

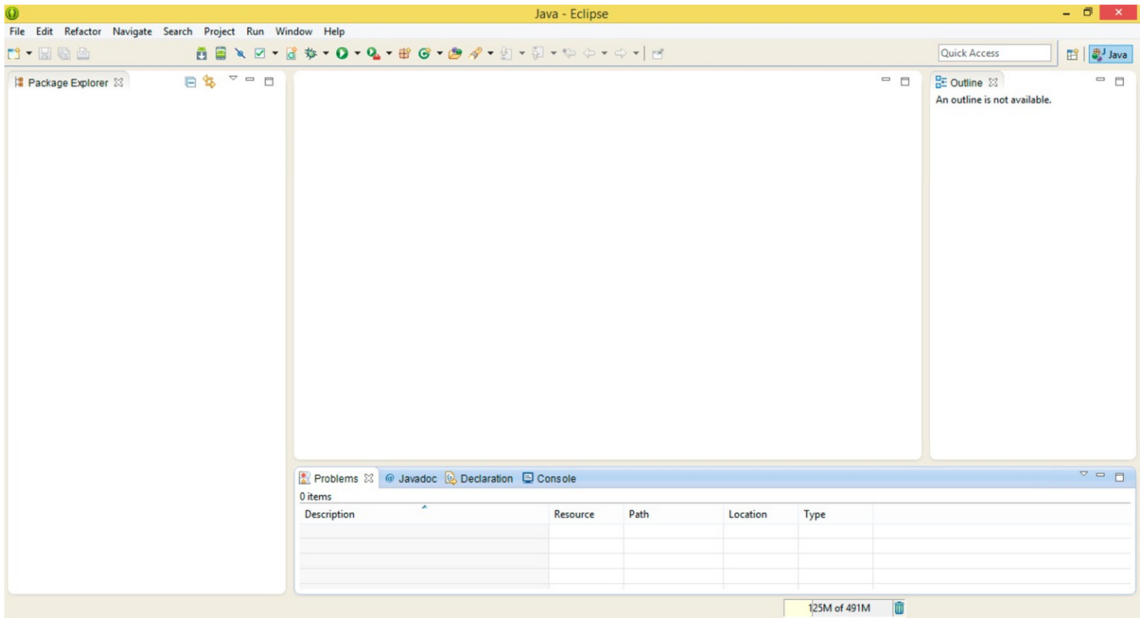


Lo primero de todo es instalar "jdk-8u20-windows-x64.exe".

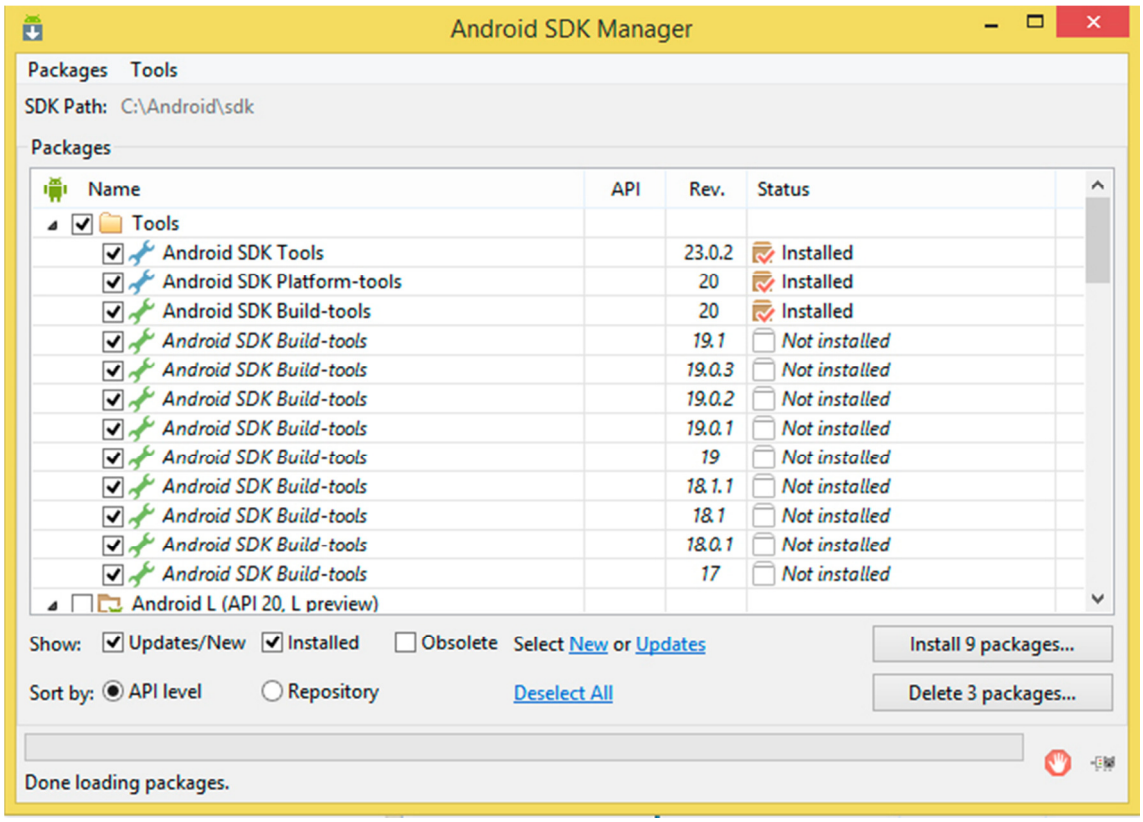
Una vez instalado, procedemos a descomprimir el fichero llamado **"adt-bundle-windows-x86-64-20140702.zip".** Una vez

descomprimido, abrimos la carpeta llamada “eclipse” y ejecutamos el fichero “eclipse.exe”.

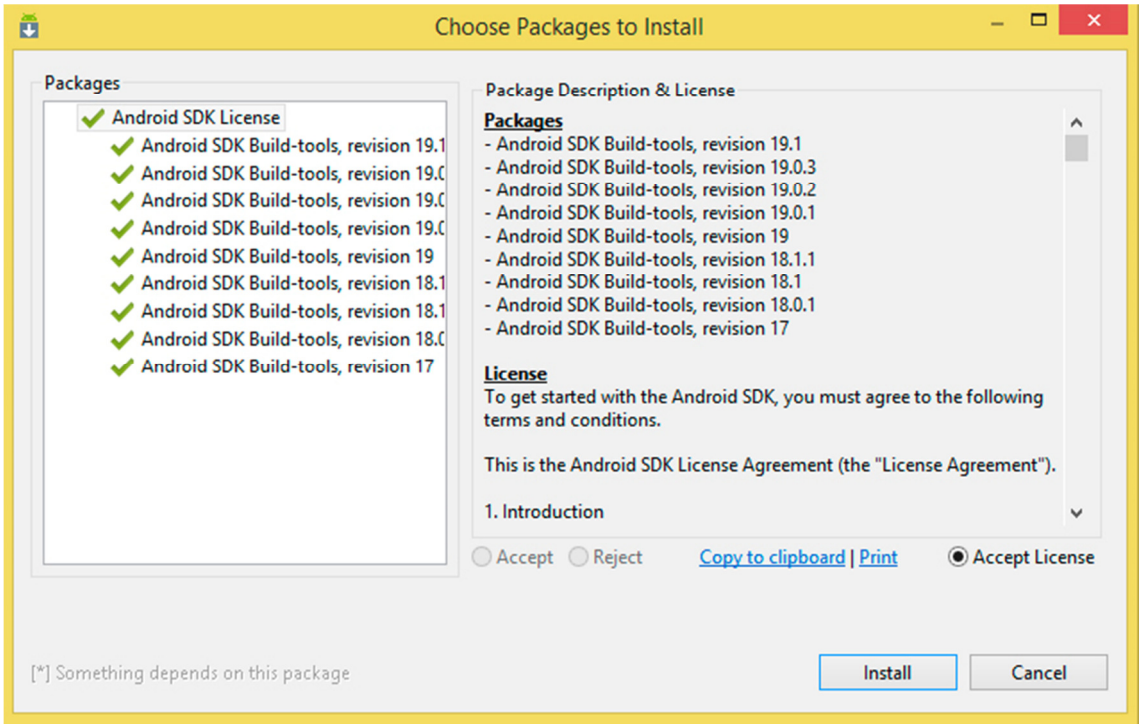
Nos pedirá un directorio llamado **“workspace”** para guardar los ficheros de los proyecto de Android. Seleccionamos el directorio a nuestra elección. Mostramos una imagen del entorno de programación “eclipse”.



Lo siguiente que debemos hacer, es proceder a la descarga de los paquetes de Android para poder compilar y ejecutar aplicaciones Android con el entorno. Para ello, buscamos el icono que aparece aquí a la izquierda en la barra de menú de arriba. Pulsamos sobre él y nos saltará una ventana con el Android SDK Manager.



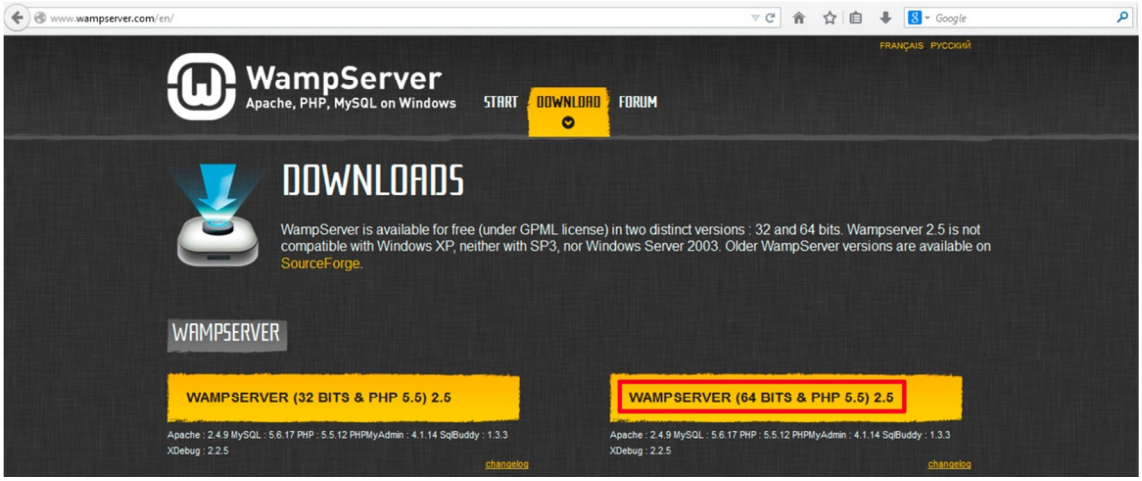
Seleccionaremos los paquetes necesarios para nuestro proyecto. Aceptaremos los términos de contrato y pulsamos instalar.



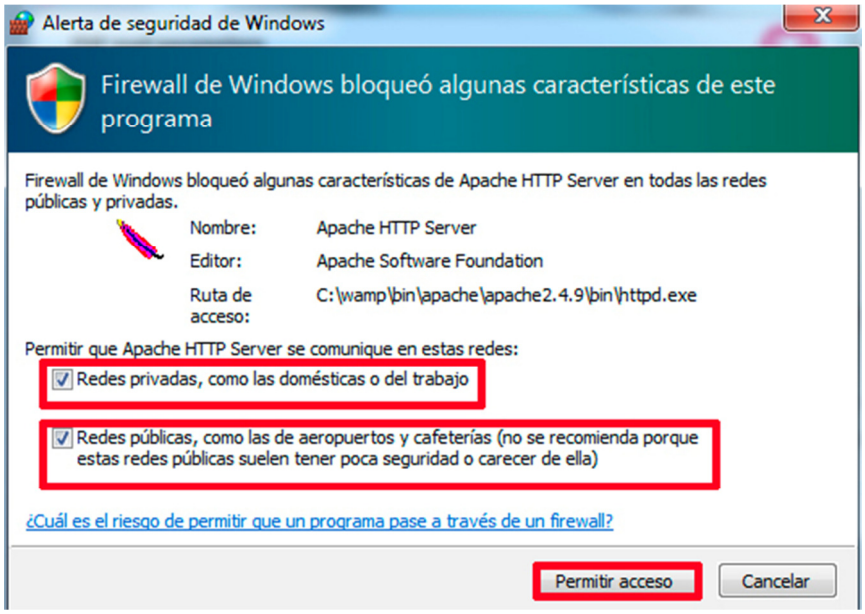
Con esto, hemos terminado la instalación y configuración del entorno de programación eclipse.

3.2. Instalación y configuración de Wamp.

En nuestro navegador, escribimos la siguiente dirección: http://www.wampserver.com y arriba nos aparecerá el botón de **"download"**, pulsamos sobre él. Debemos buscar y descargar la versión de 64 bits de WampServer.

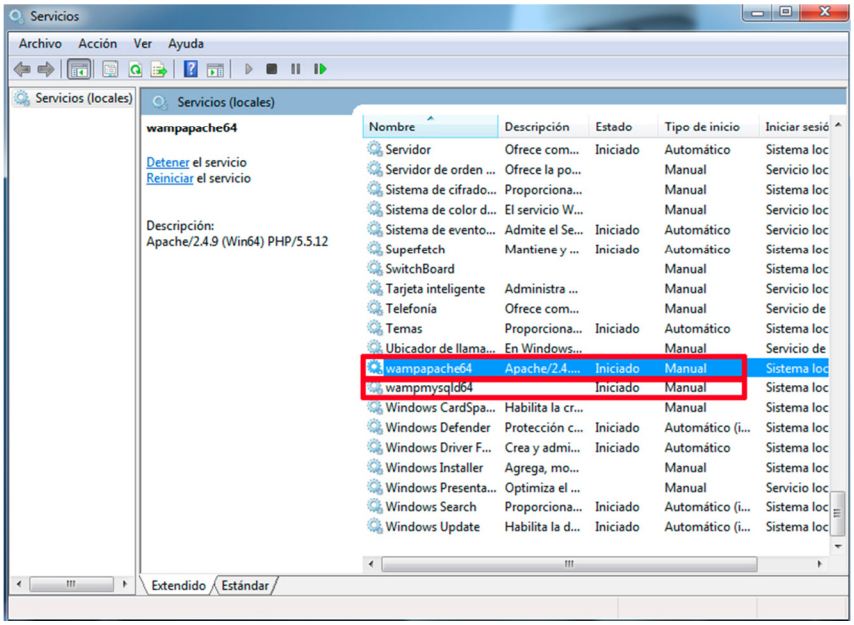


Una vez que tenemos los ficheros descargados, procedemos a su instalación. Buscamos el fichero llamado **"wampserver2.5-Apache-2.4.9-Mysql-5.6.17-php5.5.12-64b.exe"** y hacemos doble clic sobre él. Probablemente nos aparecerá en pantalla una alerta del firewall de Windows, pulsamos **"Permitir Acceso"**.



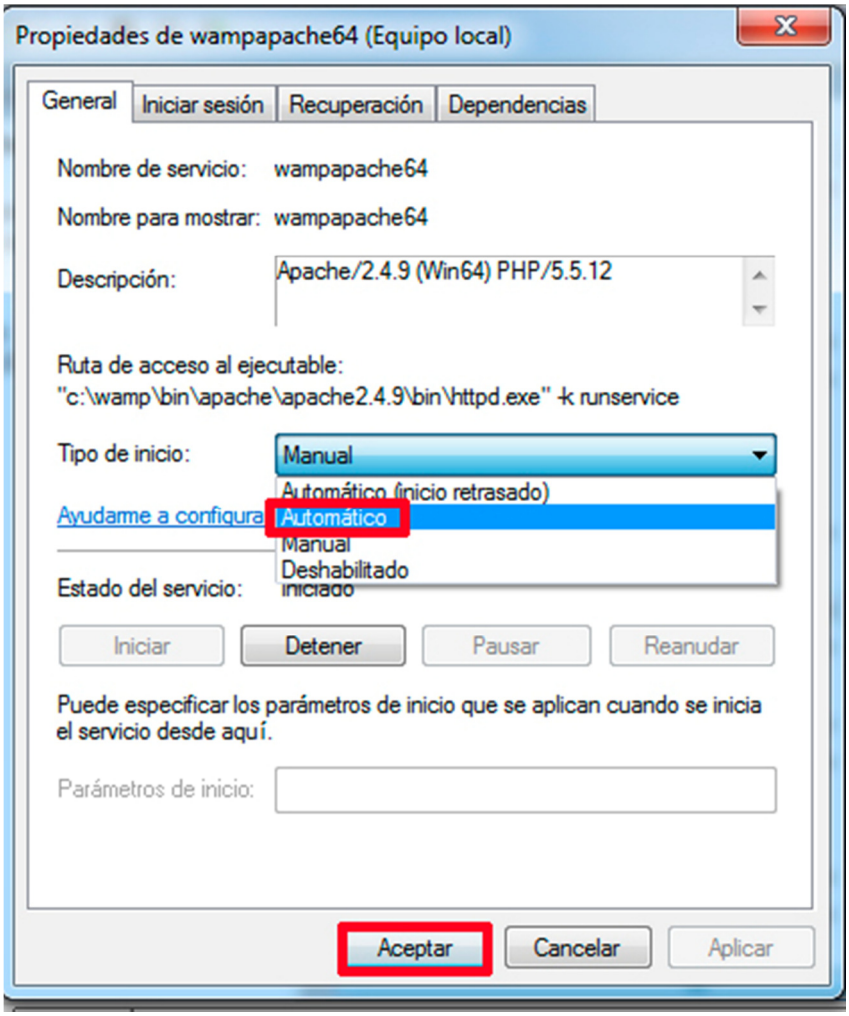
Con esto, hemos terminado la instalación. Wamp no tiene levantados por defecto los **"Servicios"**. Para que carguen automáticamente cuando iniciamos Windows, debemos ir a **"Panel de Control", "Herramientas Administrativas"** y hacemos doble clic en

“Servicios”. Buscaremos los servicios de wamp cuyos nombres son los que aparecen en la siguiente imagen:



Hacemos doble clic en **“wampapache64”** y configuramos el servicio tal y como aparece en la imagen de abajo:

Tipo de inicio: “Automatico” y aceptar.



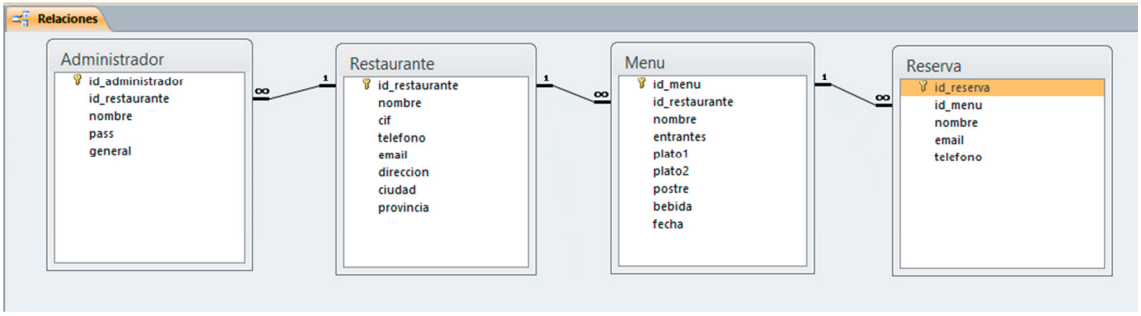
De igual forma, repetiremos estos dos últimos pasos con el servicio llamado **"wampmysqld64"**.

3.3. Creación de las Tablas en la base de datos Mysql.

La base de datos (BBDD) está compuesta por cuatro tablas que son las siguientes:

- **Administrador:** esta tabla guarda nombre, contraseña y tipo de administrador.
- **Restaurantes:** almacena el nombre, cif, teléfono, email, dirección, ciudad y provincia.
- **Menús:** esta tabla guarda la información del nombre, entrantes, primer y segundo plato, bebida, postre y fecha.
- **Reservas:** esta tabla almacena el nombre, email y teléfono.

En la siguiente imagen, se muestran los distintos atributos que tienen cada una de las tablas generadas y además se muestran las relaciones que intervienen en cada una de las tablas:



Como podemos observar, en nuestro caso, las tablas tienen cardinalidad **1:N** ó **N:1**. A continuación, mostramos la definición general de cada una de estas relaciones:

1:N: Un registro en una entidad en A se relaciona con uno o muchos registros en una entidad B pero los registros de B solamente se relacionan con un registro en A.

N:1: Una entidad en A se relaciona exclusivamente con una entidad en B pero una entidad en B se puede relacionar con uno o muchas entidades en A

Para la creación y configuración de la base de datos, abriremos un navegador y en él introduciremos la siguiente dirección: http://localhost/phpmyadmin
Introducimos los datos siguientes:

Usuario: “root”
Contraseña: “”

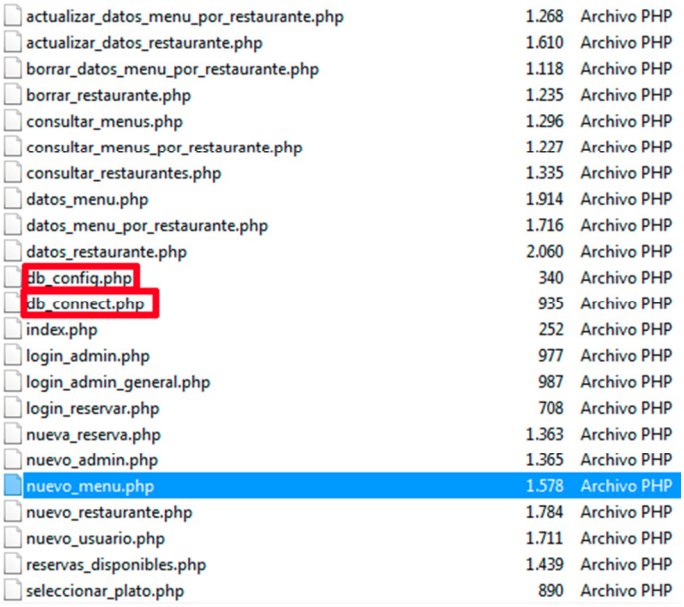
A continuación, procederemos con la creación de una base de datos llamada **“android”**. Finalmente cargamos el fichero con todas las instrucciones sql que contienen la información necesaria para crear las tablas para su correcto funcionamiento.

Una vez cargado el fichero con información de la creación de tablas, deberán aparecer como en la siguiente captura.

3.4. Configuración PHP.

Dentro del directorio de wamp, cuya ruta es “C:\wamp\” hay un directorio llamado “www”. Dentro de este últimos vamos a crear una carpeta llamada “pfc” donde vamos a guardar todos los ficheros php que van a interactuar con la base de datos, por ejemplo, insertar un menú, seleccionar las reservas disponibles, etc.

Mostramos los ficheros php que tenemos creados.



Los ficheros más importantes son los que están marcados en rojo, ya que son los que tienen la configuración de acceso de usuario y contraseña para la base de datos y tienen las funciones declaradas para conectar o cerrar la conexión a la base de datos, entre otras ellas.

A continuación, mostramos el código php que tienen cada uno de ellos.

El fichero “db_config.php” contiene los datos necesarios para poder establecer la conexión con la base de datos:

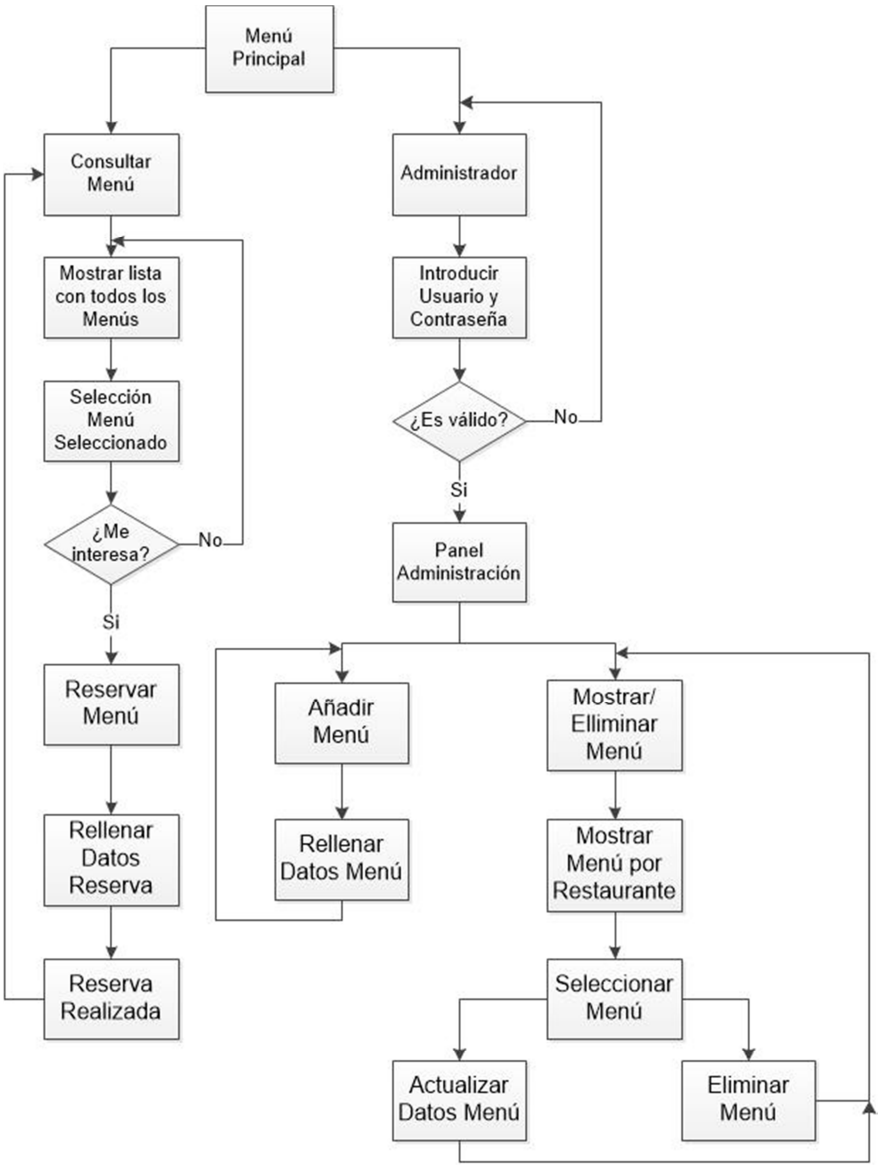
```php
<?php

    define('DB_USER', "sbernabe");
    define('DB_PASSWORD', "****************");
    define('DB_DATABASE', "android");
    define('DB_SERVER', "localhost");
?>
```

A continuación, se muestra el fichero “db_connect.php”, en el podemos observar que contiene las funciones de conectar, desconectar de la base de datos.

```php
<?php

class DB_CONNECT {

  // constructor
  function __construct() {
    // connecting to database
    $this->connect();
  }

  // destructor
  function __destruct() {
    // closing db connection
```

```
                $this->close();
        }

function connect() {

        // import database connection variables

        require_once __DIR__ . '/db_config.php';

        // Connecting to mysql database

        $con = mysql_connect(DB_SERVER, DB_USER, DB_PASSWORD) or die(mysql_error());

        $db = mysql_select_db(DB_DATABASE) or die(mysql_error()) or die(mysql_error());

        return $con;
}

 function close() {
        // closing db connection
         mysql_close();
}

}
?>
```

3.5. Diagrama de la aplicación.



En este diagrama se muestra cómo funciona la aplicación, como interactúa entre ella y los caminos para seguir avanzando y por el contrario, el camino de retorno.

4. Programando la Aplicación.

4.1. Android Manifest.

PFC Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.pfc"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="10" />
    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.pfc.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="LoginError"></activity>
        <activity android:name="LoginAdministrador"></activity>
        <activity android:name="ConsultarMenus"></activity>
        <activity android:name="ConsultarMenusPorRestaurante"></activity>
        <activity android:name="ConsultarRestaurantes"></activity>
        <activity android:name="Administrador"></activity>
        <activity android:name="AdministradorGeneral"></activity>
        <activity android:name="LoginReservar"></activity>
        <activity android:name="NuevoUsuario"></activity>
        <activity android:name="NuevoRestaurante"></activity>
        <activity android:name="NuevoMenu"></activity>
        <activity android:name="NuevoAdmin"></activity>
        <activity android:name="ConsultarMenusDatos"></activity>
        <activity android:name="ConsultarRestaurantesDatos"></activity>
        <activity android:name="ConsultarMenusPorRestauranteDatos"></activity>
        <activity android:name="ReservarMenu"></activity>
        <activity android:name="ReservasDisponibles"></activity>

    </application>

</manifest>
```

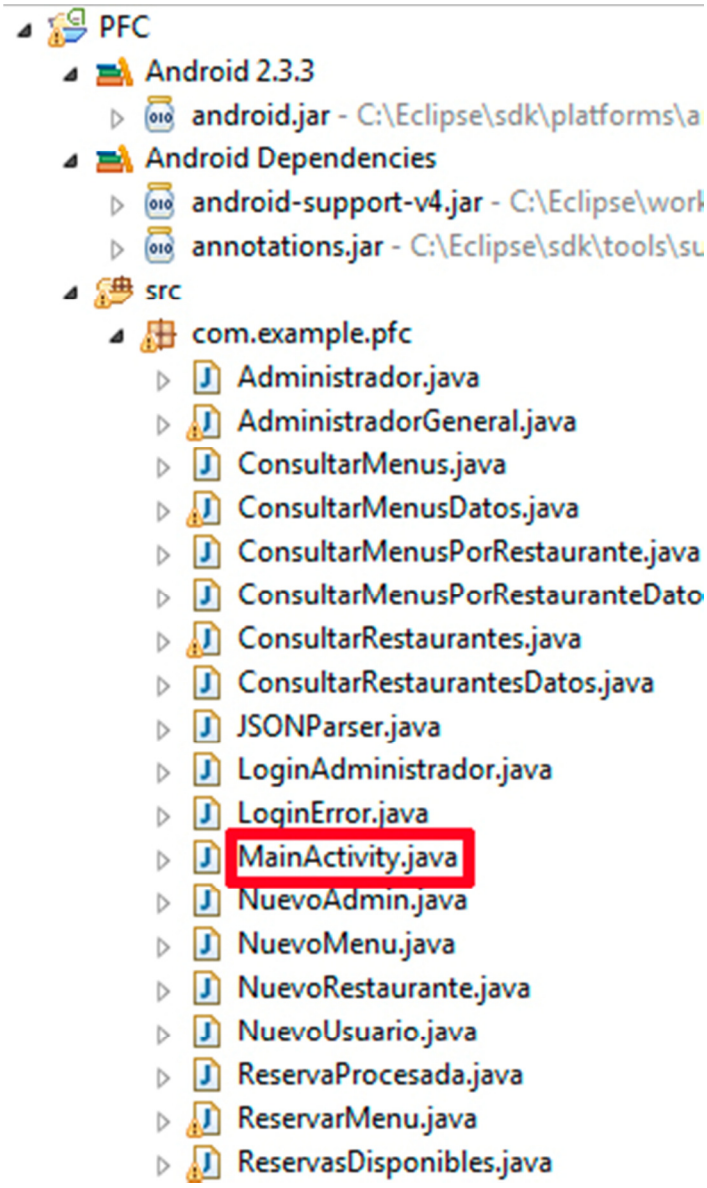
En una aplicación habitual, dentro de este archivo habrá un elemento <application>, dentro del cual habrá uno o varios elementos <activity>. Cada uno de estos elementos supone una interacción con el usuario que generalmente contiene una ventana y se corresponde con una clase que hereda de la clase Activity.

Además, en este archivo, damos permisos para usar acceso a "internet" con la aplicación mediante el contenedor llamado "uses-permission".

Cada activity está conformada por una parte lógica (un archivo Java) y una parte gráfica (un archivo XML). El archivo XML tiene todos los elementos que estamos viendo en la pantalla, declarados con etiquetas parecidas a las de HTML. El archivo .java es la clase, que hereda de la clase Activity, que se crea para poder manipular, interactuar y colocar el código de ese activity. En este archivo esta implementado todo el código relacionado con la interacción con la pantalla. Por ejemplo, cuando pulsamos un botón, lo detecta y ejecuta la acción correspondiente, como llevarte a la siguiente pantalla.

4.2. Clases Java.

A continuación, se muestra en la imagen todas las clases creadas en la aplicación.



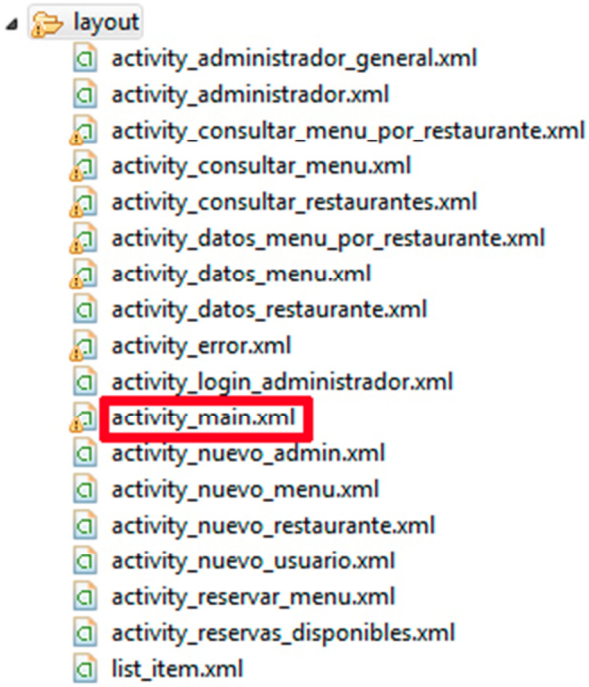
Cabe resaltar la clase principal, MainActivity.java que se encuentra resaltada en rojo. Cuando lanzamos la aplicación esta es la clase que ejecuta el menú principal de la aplicación.

Todas las demás clases se corresponden, como su propio nombre, indica con la función que realizan. Además, cada una de estas clases está asociada a un activity.

El código de todas ellas está incluido en CD de la aplicación.

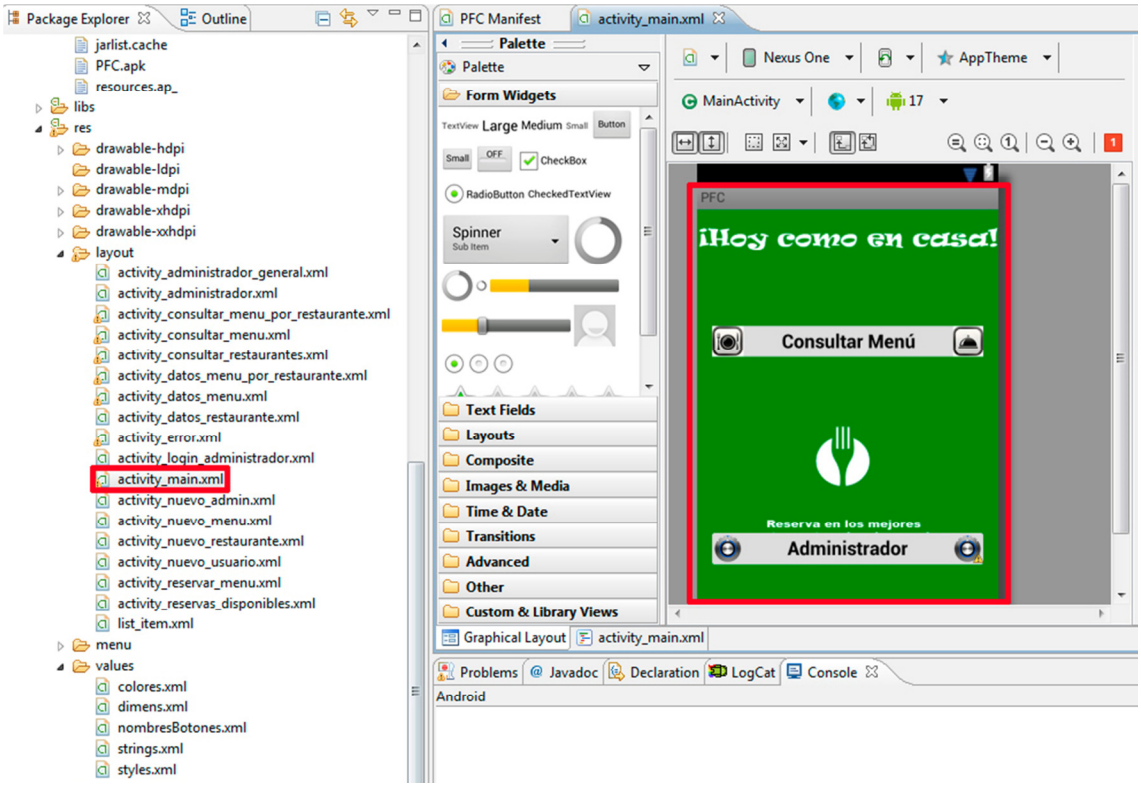
4.3. Listado de Archivos XML de la Aplicación.

En el apartado anterior se han listado todas las clases asociadas a los activitys. En la figura siguiente mostramos los archivos xml correspondientes.



Un ejemplo de archivo xml es el archivo "activity_main.xml", asociado a la clase MainActivity.java.

En la imagen de abajo, se puede observar cómo se ha construido la interfaz eligiendo los componentes para su desarrollo, como pueden ser botones y cuadros de texto entre otras opciones. Todo ello está disponible dentro del menú Palet.



4.4. Ejemplo de Activity: añadir un Nuevo Menú.

A modo de ejemplo vamos a mostrar el activity que permite añadir un nuevo menú. Este activity tiene asociados los archivos "activity_nuevo_menu.xml" y "NuevoMenu.java".

Ambos archivos contienen los comentarios necesarios para entender su funcionalidad.

El fichero activity_nuevo_menu.xml contiene la siguiente estructura de código:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/verde11"
        android:orientation="vertical" >

  <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Añadir Nuevo Menú"
        android:textAppearance="?android:attr/text
                AppearanceMedium"  />

<!--Definimos un elemento TableLayout. Dentro de
él vamos creando TableRow (columnas) y vamos
añadiendo los elementos TextView (un ejemplo
puede ser Nombre) y después en la misma fila
creamos un EditText (cuadro de texto que se edita
cuando se ejecuta la aplicacion)
Muy importante aquí es saber cuándo abre/cierra
cada elemento -->

  <TableLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

    <TableRow
        android:id="@+id/tableRow1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
```

```
                android:layout_height="wrap_content"
                android:text="Nombre:"
                android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
                android:id="@+id/editText1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:ems="10" >

        </EditText>

    </TableRow>

    <TableRow
          android:id="@+id/tableRow2"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content" >

         <TextView
                 android:id="@+id/textView3"
                 android:layout_width="wrap_content"
                 android:layout_height="wrap_content"
                 android:text="Entrantes:"
                 android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
                 android:id="@+id/editText2"
                 android:layout_width="wrap_content"
                 android:layout_height="wrap_content"
                 android:ems="10" />

    </TableRow>

    <TableRow
          android:id="@+id/tableRow3"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content" >

          <TextView
                 android:id="@+id/textView4"
                 android:layout_width="wrap_content"
                 android:layout_height="wrap_content"
                 android:text="1º Plato:"
                 android:textAppearance="?android:attr/textAppearanceMedium" />

          <EditText
                  android:id="@+id/editText3"
                 android:layout_width="wrap_content"
                 android:layout_height="wrap_content"
```

```
                android:ems="10" />

        </TableRow>

        <TableRow
            android:id="@+id/tableRow4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <TextView
                    android:id="@+id/textView5"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="2º Plato:"
                    android:textAppearance="?android:attr/textAppearanceMedium" />

              <EditText
                    android:id="@+id/editText4"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:ems="10" />

         </TableRow>

         <TableRow
            android:id="@+id/tableRow5"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <TextView
                    android:id="@+id/textView6"
                     android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Postre:"
                    android:textAppearance="?android:attr/textAppearanceMedium" />

            <EditText
                    android:id="@+id/editText5"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                     android:ems="10" />

          </TableRow>

          <TableRow
            android:id="@+id/tableRow6"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <TextView
```

```
                android:id="@+id/textView7"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Bebida:"
                android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
                android:id="@+id/editText6"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:ems="10" />

</TableRow>

<TableRow

    android:id="@+id/tableRow7"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" >

    <TextView
                android:id="@+id/textView8"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Fecha:"
                android:textAppearance="?android:attr/textAppearanceMedium" />

    <EditText
                android:id="@+id/editText7"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:ems="10"
                android:inputType="date" />

</TableRow>

<TableRow

        android:id="@+id/tableRow8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >

        <Button
                android:id="@+id/button1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Cancelar" />

        <Button
                android:id="@+id/button2"
                android:layout_width="wrap_content"
```

```
                android:layout_height="wrap_content"
                android:text="Añadir Menú" />

        </TableRow>

    </TableLayout>

</LinearLayout>
```

A continuación, mostramos el contenido del fichero NuevoMenu.java:

```
package com.example.pfc;

//Importamos los objetos que necesitamos
import java.util.ArrayList;
import java.util.List;

import org.apache.http.NameValuePair;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONException;
import org.json.JSONObject;
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class NuevoMenu extends Activity implements OnClickListener{

//Declaracion de los componentes, variables, etc
private Button aceptar;
private Button cancelar;

private EditText nombre;
private EditText entrantes;
private EditText plato1;
private EditText plato2;
private EditText postre;
private EditText bebida;
private EditText fecha;

//Inicialización de las variables
```

```
private  int id_restaurante = 0;

  private ProgressDialog pDialog;

  JSONParser jsonParser = new JSONParser();

  String ruta = "http://www.sbernabe.es/pfc/nuevo_menu.php";

  protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_nuevo_menu);

            aceptar = (Button) findViewById(R.id.button2);
            cancelar = (Button) findViewById(R.id.button1);

            nombre = (EditText) findViewById(R.id.editText1);
            entrantes = (EditText) findViewById(R.id.editText2);
            plato1 = (EditText) findViewById(R.id.editText3);
            plato2 = (EditText) findViewById(R.id.editText4);
            postre = (EditText) findViewById(R.id.editText5);
            bebida = (EditText) findViewById(R.id.editText6);
            fecha = (EditText) findViewById(R.id.editText7);

            aceptar.setOnClickListener(this);
            cancelar.setOnClickListener(this);

            id_restaurante = getIntent().getExtras().getInt("id_restaurante");
  }
  /*Cuando hacemos clic en un botón, Android lo detecta y realizamos las operaciones
  pertinentes*/
  public void onClick(View v){

            if (aceptar == v){
                      //Llamamos a la clase CreaMenuNuevo y la ejecutamos
                      new CrearMenuNuevo().execute();
            }
            if(cancelar == v){
                      /*Por el contrario, cambiamos a otro activity y le pasamos el valor
                      id_restaurante para que realice el select a partir de dicho id
                      */
                      Intent in = new Intent(NuevoMenu.this, Administrador.class);
                      in.putExtra("id_restaurante", id_restaurante);
                      startActivity(in);
            }
            }

  class CrearMenuNuevo extends AsyncTask<String, String, String> {

            protected void onPreExecute() {
                      super.onPreExecute();
```

```
                pDialog = new ProgressDialog(NuevoMenu.this);
                pDialog.setMessage("Añadiendo Nuevo Menu en la BBDD...");
                pDialog.setIndeterminate(false);
                pDialog.setCancelable(true);
                pDialog.show();
        }

        protected String doInBackground(String... args) {

        List<NameValuePair> datos = new ArrayList<NameValuePair>();

        /*Creamos una lista llamada datos, en la cual vamos añadiendo uno a uno
todos los datos que hemos introducido en el EditText. El primer campo corresponde
con el nombre la la base de datos el segundo, se obtiene del EditText utilizando
getText() y tenemos que convertirlo a cadena por si se introduce números u otros
caracteres, para ello utilizamos toString()*/

        datos.add(new BasicNameValuePair("nombre", nombre.getText().toString()));
        datos.add(new BasicNameValuePair("entrantes",
                                        entrantes.getText().toString()));
        datos.add(new BasicNameValuePair("plato1", plato1.getText().toString()));
        datos.add(new BasicNameValuePair("plato2", plato2.getText().toString()));
        datos.add(new BasicNameValuePair("postre", postre.getText().toString()));
        datos.add(new BasicNameValuePair("bebida", bebida.getText().toString()));
        datos.add(new BasicNameValuePair("fecha", fecha.getText().toString()));
        datos.add(new BasicNameValuePair("id_restaurante", id_restaurante+""));

        // getting JSON Object

        JSONObject json = jsonParser.makeHttpRequest(ruta, "POST", datos);

        // Imprimimos lo que nos devuelve Json
        Log.d("Create Response", json.toString());

        // check for success tag
        try {
        /*Procesamos el json recibido al insertar los datos en la base de datos, si
recibimos 0, es porque no es correcto y no se ha insertado correctamente, por
el contrario, si recibimos 1 quiere decir que se ha añadido correctamente, por
tanto, cambiamos de activity*/

                int success = json.getInt("success");

                if (success == 1) {

                        // Creamos un objeto Intent para cambiar de Ac
                        Intent i = new Intent(getApplicationContext(),
                                                Administrador.class);
                        i.putExtra("id_restaurante", id_restaurante);
                        startActivity(i);
```

```
                        // closing this screen

                        finish();
                    }


                    } catch (JSONException e) {
                        e.printStackTrace();
                    }

                    return null;
                }

            protected void onPostExecute(String file_url) {
                pDialog.dismiss();
                }

            }

}
```

Ya por último, el fichero nuevo_menu.php es el que inserta los datos en la base de datos, su contenido es el siguiente:

```
<?php

    // array for JSON response
    $response = array();

    if (isset($_POST['nombre']) && isset($_POST['fecha'])) {

        $nombre = $_POST['nombre'];
        $entrantes = $_POST['entrantes'];
        $plato1 = $_POST['plato1'];
        $plato2 = $_POST['plato2'];
        $postre = $_POST['postre'];
        $bebida = $_POST['bebida'];
        $fecha = $_POST['fecha'];
        $id_restaurante = $_POST['id_restaurante'];

        require_once __DIR__ . '/db_connect.php';

        // connecting to db
        $db = new DB_CONNECT();
```

```
        // mysql inserting a new row
        $result = mysql_query("INSERT INTO menu (id_restaurante, nombre,
entrantes, plato1, plato2, postre, bebida, fecha)
         VALUES('$id_restaurante', '$nombre', '$entrantes', '$plato1', '$plato2',
'$postre', '$bebida', '$fecha')");

        if ($result) {
                // successfully inserted into database
                $response["success"] = 1;
                $response["message"] = "Nuevo Menú Creado en la BBDD";

                // echoing JSON response
                echo json_encode($response);
         } else {
                // failed to insert row
                $response["success"] = 0;
                $response["message"] = "Error, No se ha insertado en la
                                        BBDD";

                echo json_encode($response);
         }
   } else {
         $response["success"] = 0;
         $response["message"] = "Faltan Campos por rellenar";

         echo json_encode($response);
   }
?>
```

5. Manual de usuario de la Aplicación.

5.1. Funcionamiento General. Pantalla Principal.

La pantalla de inicio de la aplicación ofrece dos opciones:

- La opción “Consultar Menú”.
- La opción “Administrador”.

Mediante la primera opción, el usuario general de la aplicación podrá consultar la lista de restaurantes que publicitan sus menús a través de la aplicación.

La opción “Adminstrador” permitirá a los restaurantes acceder a la aplicación para gestionar (añadir, modificar o eliminar) la publicación de las diferentes Menús.

5.2. Consultar Menú.

En este apartado, el usuario puede ver las distintas cartas de Menús proporcionadas por los distintos restaurantes.

Para ello, lo primero que debemos hacer es elegir el restaurante, como se puede observar en la imagen de la derecha.

Posteriormente, se muestra el listado de los menús ofertados por dicho restaurante, indicando la fecha del día correspondiente (imagen inferior de la izquierda).

Si seleccionamos un menú en particular, podemos ver los detalles de composición del mismo (imagen inferior a la derecha).

5.3. Reservar Menú.

Como se indica en la sección anterior, la aplicación “Hoy como en casa” permite a los usuarios hacer una reserva de un menú determinado.

Para realizar una reserva en nuestro sistema deberemos introducir la siguiente información de contacto:

- Nombre.
- Email de contacto.
- Teléfono de contacto.

Esta información permitirá al restaurante contactar con el usuario en caso de alguna incidencia con su reserva.

5.4. Modo Administrador.

La persona encargada, por parte del restaurante, de añadir, modificar o eliminar menús deberá acceder a la aplicación a través del botón Administrador de la pantalla principal de la aplicación.

Una vez identificado, el encargado del restaurante se encontrará con una pantalla como la mostrada en la figura de la izquierda.

Las opciones ofrecidas son:

- Añadir nuevo menú.
- Modificar o eliminar un menú existente.
- Consultar las Reservas realizadas.
- Abandonar el Modo Administrador.

Si pulsamos el botón de Reservas Solicitadas nos aparecen todas las reservas con su nombre, el día de la reserva y el teléfono de contacto del cliente que ha solicitado la reserva tal y como se ve en la captura de la izquierda.

5.5. Administrador General de la Aplicación.

La inclusión de nuevos restaurantes en la aplicación será llevada a cabo por el administrador de la aplicación.

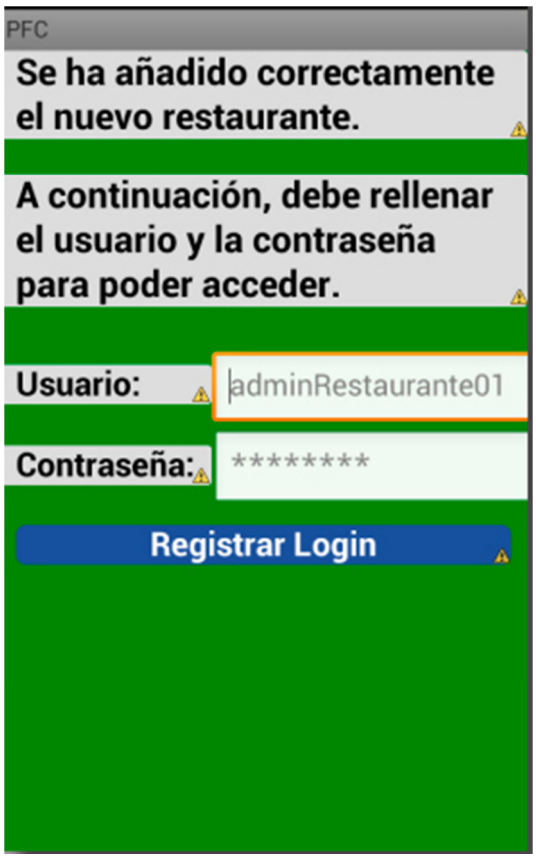
Tras identificarse, el administrador de la aplicación se encontrará con una pantalla como la mostrada en la figura de la derecha.

Las opciones ofrecidas son las siguientes:

- Añadir nuevo restaurante.
- Modificar o eliminar un restaurante existente.

Para añadir un nuevo restaurante se deberán rellenar los datos básicos del mismo, como se puede apreciar en la figura de la izquierda.

Una vez añadido dicho restaurante, se asociará a dicho restaurante un nombre de usuario y una contraseña, como muestra la figura de la derecha. Este nombre de usuario y esta contraseña serán las utilizadas por el responsable del restaurante en cuestión para acceder al modo Administrador y poder interactuar con la aplicación como se ha explicado anteriormente.



Si se pulsa el botón de Editar / Eliminar, aparecerán una lista con todos los distintos restaurantes dados de alta en la aplicación.

Desde esa lista se puede acceder al restaurante en particular para modificar sus datos o simplemente para eliminarlo. Dichas opciones corresponden con las siguientes imágenes.

6. Conclusiones y líneas futuras.

Aunque en este trabajo hemos obtenido versión completa y funcional de la aplicación, hemos identificado algunas mejoras y funciones adicionales que podrían incluirse en versiones posteriores de la app “Hoy como en casa”.

Desde el punto de vista de la programación, hemos identificado algunas mejoras que permitirían obtener una aplicación más eficaz y completa:

- Permitir a los restaurantes asociar fotos de los platos que componen sus menús. Esta funcionalidad implica modificaciones tanto en el diseño de la base de datos como en la propia aplicación.
- Modificación de la base de datos para permitir la reutilización en diferentes menús de un restaurante de los platos que se repiten a lo largo del tiempo.
- Encriptación de contraseñas en la base de datos y de todos de aquellos datos necesarios que sean necesarios.

En la actualidad, una aplicación móvil tiene que estar presente con las tecnologías que piden los usuarios, es decir, que la aplicación tenga un estilo social (web 2.0). Para ello, las mejoras pueden ser las siguientes:

- El usuario pueda poner una puntuación de 0 a 5 por los servicios prestados, calidad, etc.
- Guardas las posiciones de los diferentes restaurantes y compartir su ubicación de como poder llegar a él.
- Poder realizar una ruta por cada ciudad con los diferentes restaurantes.

7. Bibliografía.

Professional Android™ 2 Application Development; Reto Meier

Referencias Webs.

http://developer.android.com/index.html

http://stackoverflow.com/

http://www.sgoliver.net/

http://www.w3schools.com/

http://www.definicionabc.com/tecnologia/mysql.php

http://json.org/json-es.html

http://www.ajaxya.com.ar/temarios/descripcion.php?cod=26&punto=19

http://es.wikipedia.org/wiki/PHP

http://www.codejobs.biz/es/blog/2013/11/06/descripcion-general-de-php#sthash.WnKJSygx.dpbs

http://definicion.de/php/

http://paletton.com/